A review on methods and software for fuzzy cognitive maps

Peer-reviewed author version

Felix, Gerardo; NAPOLES RUIZ, Gonzalo; Falcon, Rafael; Froelich, Wojciech; VANHOOF, Koen & Bello, Rafael (2017) A review on methods and software for fuzzy cognitive maps. In: Artificial intelligence review, 52 (3), p. 1707-1737.

DOI: 10.1007/s10462-017-9575-1
Handle: http://hdl.handle.net/1942/25510

Artificial Intelligence Review manuscript No.
(will be inserted by the editor)

A Review on Methods and Software for Fuzzy Cognitive Maps

Gerardo Felix · Gonzalo Nápoles · Rafael Falcon · Wojciech Froelich · Koen Vanhoof · Rafael Bello

Received: date / Accepted: date

Abstract Fuzzy Cognitive Maps (FCMs) keep growing in popularity within the scientific community. However, despite substantial advances in the theory and applications of FCMs, there is a lack of an up-to-date, comprehensive presentation of the state-of-the-art in this domain. In this review study we are filling that gap. First, we present basic FCM concepts and analyze their static and dynamic properties, and next we elaborate on existing algorithms used for learning the FCM structure. Second, we provide a goal-driven overview of numerous theoretical developments recently reported in this area. Moreover, we consider the application of FCMs to time series forecasting and classification. Finally, in order to support the readers in their own research, we provide an overview of the existing software tools enabling the implementation of both existing FCM schemes as well as prospective theoretical and/or practical contributions.

Keywords Fuzzy Cognitive Maps · Machine Learning · Software Tools

Gerardo Felix
Central University of Las Villas, Cuba
Tel.: +53-42-281515
E-mail: gerardfelix87@gmail.com

Gonzalo Nápoles
Hasselt University, Belgium
E-mail: gonzalo.napoles@uhasselt.be

Rafael Falcon
University of Ottawa, Canada
E-mail: rfalcon@uottawa.ca

Koen Vanhoof
Hasselt University, Belgium
E-mail: koen.vanhoof@uhasselt.be

Wojciech Froelich
University of Silesia, Poland
E-mail: wojciech.froelich@us.edu.pl

Rafael Bello
Central University of Las Villas, Cuba
E-mail: rbellop@uclv.cu

1 Introduction

Fuzzy Cognitive Maps were proposed by Kosko [48] as a graph-based knowledge representation method describing a set of concepts in a domain of interest that are connected by cause-and-effect relationships among them.

From the structural perspective, an FCM is a cognitive digraph that describes the behavior of a physical system in terms of nodes and edges connecting them. The concepts (i.e, nodes of the graph) can be understood as fuzzy sets describing the variables, objects or entities of the system under investigation. The signed and weighted edges of the graph represent the causal relationships among the concepts. By characterizing the interaction between fuzzy sets over several iterations, FCMs are able to represent vague and complex scenarios.

During the last thirty years of FCM research, substantial improvements to the original FCM architecture have emerged. FCMs became an important member of the Soft Computing family [128] as they were capable of efficiently solving numerous problems across a variety of domains such as decision support, system control, time series forecasting, pattern classification, and many others. Spurred by tangible progress, research on FCMs continues to draw a great deal of interest from many researchers around the globe. Recently, the *Neurocomputing* journal dedicated a Special Issue [30] to the theoretical advances of fuzzy cognitive mapping, which comprises fresh contributions to the this field.

There are numerous papers published within the FCM arena. Each of them briefly revisits the basic concepts behind FCM theory. However, most of those reviews are rather partial and often focused on the specific problem addressed in the paper. To the best of our knowledge, the latest comprehensive report on FCMs was published by Papageorgiou and Salmeron [97] in 2013. Substantial advances in the theory of FCMs have surfaced since then. In this paper we fill this gap by presenting the most up-to-date review on FCMs to the reader.

Another limitation of older FCM survey papers is the lack of practical advice about implementation issues. The gap between the theoretical advances and the development of accurate and mathematically sound FCM-based systems advocates for the proliferation of software tools with more complete experimentation features. Without knowledge on existing FCM software tools, most of the researchers are forced to develop their own implementations from scratch. In addition to being a laborious task, such implementations are hard to integrate or even compare with each other. In this paper, we outline the existing software tools that are available to implement, test and validate FCM-based models. As a more general goal, this paper attempts providing the reader a clear theoretical basis and practical support towards implementing FCM-based solutions.

The rest of this paper is organized as follows. Section 2 goes over the mathematical underpinnings of *fuzzy cognitive mapping* and its dynamic properties. Section 3 describes the most prominent algorithms for learning the map structure and lays out a comparative analysis between Hebbian-based and error-minization-driven algorithms. Section 4 elaborates on FCM-based times series forecasting while Section 5 reports on recent advances and challenges behind FCM-based classification. Section 6 reviews the existing software tools for experimenting with FCMs while the last section enunciates some concluding remarks.

2 Fuzzy Cognitive Maps

The semantics behind a standard FCM can be defined by a 4-tuple $(\mathcal{C}, \mathcal{W}, \mathcal{A}, f)$, where $\mathcal{C} = \{C_1, C_2, \ldots, C_M\}$ is the family of M concepts modeled after fuzzy sets, $\mathcal{W} : \mathcal{C} \times \mathcal{C} \rightarrow [-1, 1]$ is the matrix containing the weight $w_{ij} \in [-1, 1]$ assigned to each pair of concepts (C_i, C_j). The value of w_{ij} determines the sign and intensity (magnitude) of the edge connecting the cause concept C_i with the effect concept C_j. The function $\mathcal{A} : \mathcal{C} \rightarrow A_i^{(t)}$ computes the activation degree $A_i \in \mathfrak{R}$ of each concept C_i at the discrete time step $t = \{1, 2, \ldots, T\}$. Finally, the transfer function $f : \mathfrak{R} \rightarrow I$ aggregates the impact of multiple causal events over the target concept and clamps the result to the predefined activation interval. The interpretation of the causal weight w_{ij} between two concepts C_i and C_j is as follows:

- If $w_{ij} > 0$, then an increment (decrement) in the concept C_i will produce an increment (decrement) on concept C_j with intensity $|w_{ij}|$.
- If $w_{ij} < 0$, then an increment (decrement) in the concept C_i will produce a decrement (increment) on concept C_j with intensity $|w_{ij}|$.
- If $w_{ij} = 0$ (or very close to 0), this denotes the absence of a causal relationship from C_i upon C_j, so there is no causal edge in the graph.

Other FCM extensions are designed to achieve flexibility when modeling complex systems. As an example, Miao et al. [67] introduced the *Dynamical Cognitive Network* where each concept can have its own value set, depending on how precisely it needs to be described in the network. In these FCM-based networks, the edges on the digraph define dynamic, causal relationships among concepts. This scheme is able to capture not only the causal relationship but also how it will make the effect and how long it takes for the effect to build up.

Equation (1) displays Kosko's activation rule for FCMs, with $A^{(0)}$ being the initial configuration (activation vector), w_{ji} the value of the causal relation connecting concept C_j to concept C_i whereas $A_i^{(t)}$ denotes the activation value of concept C_i at the t-th time step. This update rule is iteratively repeated until a stop condition is met. Notice that an FCM will produce a state vector at each discrete time step that comprises the activation degree of all concepts.

$$A_i^{(t+1)} = f\left(\sum_{\substack{j=1 \\ i \neq j}}^{M} w_{ji} A_j^{(t)}\right) \tag{1}$$

The above equation describes an inference (updating) rule widely used in many FCM-based applications, but it is not the only one. Stylios and Groumpos [127] proposed a modified inference rule where concepts take into account their own past activation value besides the corresponding weights and activation values coming from other concepts. This reasoning rule is preferred when updating concepts that are not influenced by other concepts. The reader can notice that this implicitly removes the $i \neq j$ constraint in Kosko's equation.

$$A_i^{(t+1)} = f\left(\sum_{\substack{j=1 \\ i \neq j}}^{M} w_{ji} A_j^{(t)} + A_i^{(t)}\right) \tag{2}$$

Another modified update rule was proposed in [86] to avoid the conflicts emerging in the case of inactive concepts. Being more explicit, the rescaled inference depicted in Equation (3) allows dealing with the scenarios where there is no information about an initial concept state and helps prevent the saturation problem (i.e., the activation values of processing entities careen toward their minimal/maximal values as a result of a dense information flow described by similar causal signs). In some scenarios, we could counter the aforementioned issues by using the proper parametric settings in the transfer function.

$$A_i^{(t+1)} = f\left(\sum_{\substack{j=1 \\ i \neq j}}^{M} w_{ji}(2A_j^{(t)} - 1) + (2A_i^{(t)} - 1)\right) \tag{3}$$

Selecting the proper update rule depends on the problem at hand and regularly requires a strong understanding of the physical/simulated system under investigation. As a further valuable remark, Papakostas and Koulouriotis [102] observed that removing the $i \neq j$ restriction in Equations (1) and (2) does not necessarily improve the overall prediction rates of FCM-based classifiers.

One of the most relevant characteristics of FCMs is the interpretability of their topology. In fact, FCMs can be defined as *interpretable recurrent neural networks* that include fuzzy logic elements during the knowledge engineering phase. More explicitly, an FCM exploits an activation vector by using a rule similar to the standard McCulloch-Pitts model [65] where concepts can be thought of as neural processing entities. This implies that the activation degree of each map neuron is given by the value of the transformed weighted sum this processing unit receives from the connected neurons in the causal network.

A few papers in the literature (e.g., [130] [129] [103] [55] [80] [81]) construe FCM-based models as Artificial Neural Networks (ANNs), even when some FCM theoretical methods (e.g., Hebbian learning algorithms) have a clear neural meaning. The reason behind that relies on their differences. Classical ANNs regularly perform like black boxes where both hidden neurons and connections do not bear any clear meaning to the problem itself [80]. However, the FCM neurons and their connections do have a precise interpretation for the system under study. Besides, FCMs do not involve hidden neurons since such entities could neither be interpreted nor help explain why a solution is suitable for a given problem. This suggests that the representation capability of FCM-based systems is superior to that of ANN-based models, which explains why FCMs have witnessed a plethora of successful applications in modeling complex real-world scenarios.

2.1 Dynamic properties of FCM-based systems

As mentioned, FCMs produce a new state vector at each discrete time step. This procedure is repeated until either the system stabilizes or meets a predefined stop criterion (e.g., reaching a maximal number of iterations) [50]. The former implies that a hidden pattern was discovered [49], whereas the latter suggests that the FCM responses are either cyclic or completely chaotic.

If the cognitive network is able to converge, then the system will produce the same output towards the end, and therefore the activation degree of concepts will remain unaltered (or subject to infinitesimal changes). On the other hand, a cyclic FCM produces dissimilar responses with the exception of a few states that are periodically produced. The last possible scenario is related to chaotic configurations in which the network yields different state vectors. Formally, such situations are mathematically defined as follows:

- **Fixed-point** [129] ($\exists t_\alpha \in \{1, 2, \ldots, (T-1)\} : A^{(t+1)} = A^{(t)}, \forall t \geq t_\alpha$): the FCM produces the same state vector after the t_α-th iteration. This suggests that $A^{(t_\alpha)} = A^{(t_\alpha+1)} = A^{(t_\alpha+2)} = \cdots = A^{(t)}$.
- **Limit cycle** [132] ($\exists t_\alpha, P \in \{1, 2, \ldots, (T-1)\} : A^{(t+P)} = A^{(t)}, \forall t \geq t_\alpha$): the FCM periodically produces the same state vector after the t_α-th iteration. This suggests that $A^{(t_\alpha)} = A^{(t_\alpha+P)} = A^{(t_\alpha+2P)} = \cdots = A^{(t_\alpha+jP)}$ where $t_\alpha + jP \leq T$, such that $j \in \{1, 2, \ldots, (T-1)\}$.
- **Chaos** [132]: the FCM produces a different state vector at each iteration. In other words, the system is neither stable nor cyclic.

In presence of chaotic or cyclic situations, the reasoning rule stops once a maximal number of iterations T is reached. At that point, the state vector is calculated from the last response, but this output may be partially unreliable due to the lack of stability. Convergence is often desirable since the responses become consistent and the expert may understand the system behavior. However, there are scenarios lying beyond decision making and pattern classification (e.g., time series forecasting) where convergence is much less desired.

The convergence issues in FCM-based systems are mostly related to i) the pattern encoded in the weight matrix, ii) the strategy for updating the concepts' values and iii) the non-decreasing transfer function used in the reasoning rule [71]. Several studies [42] [10] [129] have shown that a symmetric zero-diagonal matrix in conjunction with an asynchronous update rule lead to improved convergence features. On the other hand, as was already explained, the transfer function and its parameters may prove crucial to reach convergence.

Boutalis et al. [14] investigated the existence and uniqueness of the fixed-point attractors in FCM-based systems equipped with sigmoid transfer functions by using the contraction mapping theorem. It was proved that when the causal weight matrix meets certain conditions, the map will converge to a unique fixed-point attractor. Besides, the authors introduced an adaptive weight estimation method that employs different weight projection criteria to guarantee that the uniqueness of the FCM solution is not compromised.

Kottas et al. [51] studied the existence and uniqueness of the equilibrium point on FCMs [52] [53]. Following the same contraction-mapping-based rationale, the authors analytically proved that when the weight matrix meets specific conditions (related to the size of the network and the sigmoid slope) the network will converge to a unique solution regardless of its initial values.

Knight et al. [46] investigated the effect of the parameter controlling the sigmoid slope over the FCM convergence. The idea is based on the assumption that should there were two stable fixed points then a slight change in initial conditions (which cannot be exactly quantified) may result in a different outcome, thus making the returned values difficult to justify. It can be observed that the model convergence does not depend on the weigh set defining the system.

In [69] [70] the authors proposed a learning algorithm to improve the system convergence without modifying the causal weights. This procedure estimates the slope parameter of each sigmoid neuron by minimizing an error function that quantifies the dissimilarity between consecutive responses for the same initial stimulus. In a deeper analysis, Nápoles and his collaborators [80] proposed four modifications related to the error function and the analytical boundaries for the search space. In the improved variant, the learning algorithm minimizes the dissimilarity between the current state vector and the expected response.

More recently, Nápoles et al. [71] put forth an extended procedure to improve the convergence features of FCM-based systems used in the simulation of multiple scenarios. In this approach, the learning algorithm simultaneously reduces the dissimilarity between two consecutive state vectors and the dissimilarity between the current state vector and the expected response. Moreover, the authors analytically investigated the trade-off between system accuracy and convergence. The numerical results reported in [71] confirmed that establishing a trade-off between accuracy and convergence cannot always be achieved without modifying the weight matrix. The empirical evidence indicates that by simply adjusting the sigmoid function parameters, some concepts are still unable to converge into an infinitesimal error region. In the case of FCM-based systems where decisions are defined by closed partitions of the activation space, this outcome may be acceptable.

2.2 The transfer function

The function $f : \Re \rightarrow I$ in Equations (1), (2) and (3) denotes a monotonically, non-decreasing function used to clamp the activation value of each concept to the desired interval I, where $I = [0, 1]$ or $I = [-1, 1]$ depending on the problem domain. According to the cardinality of the state space, existing transfer functions may be categorized into either *discrete* or *continuous*. The most widely used transfer functions are the bivalent, the trivalent, the hyperbolic tangent and the sigmoid function. Next, we explain some of their properties.

1. *The bivalent function* (see Equation (4)). It is a well-known discrete function that only produces binary responses leading to a finite number of states. This happens because an FCM is a deterministic system and so, if it reaches a state that it has previously visited, the system will enter a closed orbit which will always repeat itself [129]. Therefore, a binary FCM will either converge to a fixed-point attractor or show cyclic patterns (with an exponential period in the worst scenario) but it will never produce chaos.

$$f_1(x) = \begin{cases} 1 & , x > 0 \\ 0 & , x \leq 0 \end{cases} \tag{4}$$

2. *The trivalent function* (see Equation (5)). It is another discrete transfer function that produces a finite number of different states; therefore, the network will either converge to a fixed-point attractor or produce cyclic patterns, but chaos is not possible. The key disadvantage of discrete transfer functions lies in its poor representation capabilities given that only rather qualitative (i.e., bivalent or trivalent) scenarios can be modeled.

$$f_2(x) = \begin{cases} -1 & , x < 0 \\ 0 & , x = 0 \\ 1 & , x > 0 \end{cases} \tag{5}$$

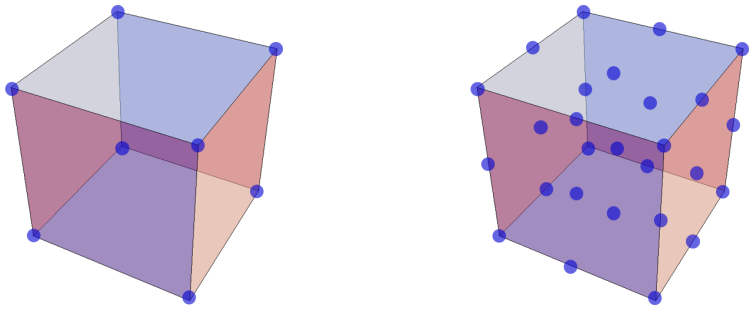
3. *The hyperbolic function* (see Equation (6)). It is a continuous transfer function that produces infinite states that are freely distributed within the $[-1, 1]^M$ hypercube. Besides the equilibrium points and the cyclic states, continuous functions might additionally yield chaotic outputs [129]. However, they can be used for modeling both qualitative and quantitative scenarios.

$$f_3(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{6}$$

4. *The sigmoid function* (see Equation (7)). It is a continuous function that produces an infinite number of different states that are freely distributed within the $[0, 1]^M$ hypercube. In this function, $\lambda > 0$ and $h > 0$ are two user-specified parameters controlling the function slope and offset, respectively. Higher values of λ increase the steepness and make it more sensitive to the fluctuations of x, hence the derivative grows as the activation value goes up.

$$f_4(x) = \frac{1}{1 + e^{-\lambda(x-h)}} \tag{7}$$

Figure 1 displays the state space of an FCM-based network with three concepts for both the bivalent and trivalent functions. In the bivalent variant, the states are located in the corners of the $[0, 1]^2$ hypercube, whereas in the case of the trivalent variant the states lie at the corners, at the middle of the edges, at the center of the sides and at the center of the $[-1, 1]^2$ hypercube. As an interesting result, Miao and Liu [66] proved that the problem of finding whether a state is reachable in a binary FCM is *non-deterministic polynomial hard*.



(a) Bivalent transfer function. (b) Trivalent transfer function.

Fig. 1: State space of FCM-based systems with three concepts using (a) a bivalent transfer function, and (b) a trivalent transfer function.

The number of patterns an FCM is capable of recognizing increases with the number of outputs that $f(.)$ produces. Regrettably, this also bumps up the risk of producing cyclic states with larger exponential periods. This risk could be mitigated (or completely suppressed) should the weight matrix meet some conditions ensuring the convergence to a fixed-point attractor. The reader can refer to the results reported in [42] [10] [14] [129] [51] [81] for further discussion.

Nevertheless, it has been proved that the FCM exhibits a greater inference capacity whenever continuous threshold-based functions are used [129]. Bueno and Salmeron [15] conducted a benchmarking study and concluded that the sigmoid function attains the highest predictive capacity among all tested transfer functions. It is worthwhile emphasizing that the selection of the threshold function is frequently conditioned by the system requirements, i.e., by the role each concept plays in modeling the system under consideration.

Some researchers use standard parameters without taking into account their effect over the cognitive model and its behavior. For example, by using the Kosko's reasoning rule, $\lambda = 1$ and $h = 0$ in Equation (7), an inactive concept will become active after the first time step. In most real-world problems, this behavior is difficult to justify to the domain expert and often affects the cognitive representation of the physical/artificial system under consideration.

3 Learning Algorithms

As a minimalist description, the main objective behind FCM learning is to derive the weight matrix $W_{(M\times M)}$ based on expert intervention, available historical data or both. Most of the existing learning approaches assume that the set of concept labels is provided a priori by the expert [87] and only the weight matrix is learned either from raw data or with the help of domain experts.

A diverse number of learning algorithms have been proposed in the literature; they are mostly based on principles coming from the field of artificial neural networks. The most important algorithms for FCM learning can be classified into three types on the basis of their underlying learning paradigm: *Hebbian-based*, *error-driven* and *hybrid learning algorithms*. Additionally, there exist other alternative learning schemes that do not fall within the three aforementioned groups. They combine other mathematical and computational theories to build FCMs and we will go over their theoretical underpinnings opportunely.

3.1 Hebbian-based approaches

Hebbian-based learning methods are *unsupervised* procedures. As such, they do not require a collection of labeled historical records, i.e., data in which the value of the decision feature(s) are previously known. The theory behind such methods was introduced by Donald Hebb in his book "*The Organization of Behavior*" [37] and later modified through multiplicative normalization by Erkki Oja [11], thus leading to the Oja learning rule. The goal of learning FCMs by using adaptive Hebbian-based methods is to yield weight matrices on the basis of experts' knowledge and to improve the accuracy of previously defined weights.

The first Hebbian-based learning algorithm, referred to as *Differential Hebbian Learning* (DHL), was presented by Dickerson and Kosko [23]. The DHL algorithm assumes that if the cause concept C_i and the effect concept C_j change their activation values simultaneously, then the causal weight w_{ij} shall be increased by a constant factor; otherwise, the causality relation will not be modified in that iteration. Equation (8) displays the weight update rule used in this algorithm, where $\Delta A_i^{(t)} = A_i^{(t)} - A_i^{(t-1)}$, with $A_i^{(t)}$ being the activation value of the C_i concept at the t-th iteration. In this formulation, the parameter η_t represents the learning rate and it is adjusted at each iteration.

$$w_{ij}^{(t+1)} = \begin{cases} w_{ij}^{(t)} + \eta_t \left(\Delta A_i^{(t)} \Delta A_j^{(t)} - w_{ij}^{(t)} \right) & , \Delta A_i^{(t)} \neq 0 \\ w_{ij}^{(t)} & , \Delta A_i^{(t)} = 0 \end{cases} \tag{8}$$

The main drawback of this local approach resides in the absence of information concerning the system as a whole. In other words, the DHL method updates the weights between each pair of concepts, thus taking into account only these two concepts and ignoring the influence coming from other concepts.

An improved DHL variant called *Balanced Differential Algorithm* (BDA) was introduced by Huerga [43] to attenuate this problem. This learning method eliminates one of the DHL limitations by taking into account all the concept values that change at the same time when the weights are updated. Specifically, it considers the change in all the concepts that are produced in the same time step and with the same directionality. BDA is a more powerful scheme than its predecessor, although not very popular (to the best of our knowledge, it has only been applied in binary pattern recognition problems, which hinders its application to other areas). Equation (9) formalizes BDA's update rule for FCMs without self-connections, which is iteratively applied over the initial weight matrix.

$$w_{ij}^{(t+1)} = \begin{cases} w_{ij}^{(t)} + \eta_t \left[\frac{\Delta A_i^{(t)} \Delta A_j^{(t)}}{\sum_{k=1}^{m} \Delta A_i^{(t)} \Delta A_j^{(t)}} - w_{ij}^{(t)} \right] & , \Delta A_i^{(t)} \Delta A_j^{(t)} > 0 \\ w_{ij}^{(t)} + \eta_t \left[\frac{-\Delta A_i^{(t)} \Delta A_j^{(t)}}{\sum_{k=1}^{m} \Delta A_i^{(t)} \Delta A_j^{(t)}} - w_{ij}^{(t)} \right] & , \Delta A_i^{(t)} \Delta A_j^{(t)} < 0 \end{cases} \tag{9}$$

Shortly after, two new Hebbian-based learning algorithms emulating synaptic plasticity, namely *Active Hebbian Learning* (AHL) and *Nonlinear Hebbian Learning* (NHL) were introduced in [85] and [91], respectively.

AHL assumes that all concepts are activated asynchronously and in this way, a fixed-point attractor is reached by considering the concepts' activation at different iterations. This mechanism is useful in systems where the concepts are activated based on a specific sequence [85]. The domain experts determine a desired set of concepts, an initial structure and interconnections of concepts as well as the sequence of activation concepts. Unlike the previous algorithms where only the nonzero weights are updated, in the AHL scheme all weights (except those pertaining to self-connections) are updated.

In the AHL learning method, concepts are categorized as either activated or activation entities, where the activation values of the former are used to activate the latter. Equation (10) displays the AHL's weight update rule, where $\check{A}_j(t)$ denotes the value of the j-th activation concept, whereas $\gamma^{(t)}$ is the weight decay

associated with the t-th iteration. Both the weight decay and the learning rate are exponentially decreased over time. The key disadvantage of this learning method lies in its reliance on expert judgment.

$$w_{ij}^{(t+1)} = \left[1 - \gamma^{(t)}\right] w_{ij}^{(t)} + \eta^{(t)} A_i^{(t)} \left[A_j^{(t)} - w_{ij}^{(t)} \check{A}_j^{(t)}\right] \tag{10}$$

NHL is another Hebbian-based learning implementation that modifies the original Hebbian learning rule. The NHL procedure initially requires the experts' intervention for determining the nature of the concepts, the range of values these concepts might take and the definition of the sign of each weighted interconnection. The initial graph structure elicited from the experts is retained during the learning process, thus preserving its physical interpretation [87]. However, this dependence on the experts' criteria becomes one of the major drawbacks of this algorithm. NHL introduces two stop criteria: (1) a close-enough solution to the desired response has been reached or (2) a fixed-point attractor has been identified. Equation (11) portrays NHL's weight update rule.

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \eta A_j^{(t)} \left[A_i^{(t)} - A_j^{(t)} w_{ij}^{(t)}\right] \tag{11}$$

As a further modification, the authors in [99] introduced a weight decay factor γ leading to the update rule depicted in Equation (12), where the inclusion of the $sgn(.)$ sign function attempts to preserve the weight directionality.

$$w_{ij}^{(t+1)} = \gamma w_{ij}^{(t)} + \eta A_j^{(t)} \left[A_i^{(t)} - \text{sgn}\left(w_{ij}^{(t)}\right) A_j^{(t)} w_{ij}^{(t)}\right] \tag{12}$$

The *Improved Nonlinear Hebbian Learning* (INHL) algorithm was proposed by Li and Shen [58]. They added a term in the update rule called the *impulse* in order to avoid getting trapped in local minima in regions where the error surface plateaus. Equation (13) shows the resultant update rule, where $\alpha \in (0, 1]$ denotes the acceleration while descending the error surface.

$$w_{ij}^{(t+1)} = \alpha^{(t+1)} \Delta w_{ij}^{(t)} + \eta^{(t+1)} \left(z^{(2t)}\right) \left[1 - z^{(t)}\right] \left[A_j^{(t)} - A_i^{(t)} w_{ij}^{(t)}\right] \tag{13}$$

where

$$z(t) = \frac{1}{1 + e^{-A_i^{(t)}}} \tag{14}$$

Later, an improved version of the NHL method named *Data-Driven NHL* (DD-NHL) was brought forth by Stach et al. [122]. The authors' main motivation relies on the fact that the NHL algorithm does not exploit any additional information that could improve learning and generate more accurate models. DD-NHL uses the same principle behind Hebbian-based procedures but instead of generating data used for learning only from the current model, it takes advantage of data available for a given system. Unlike other Hebbian-type algorithms, in this method the initial weight matrix can be randomly generated. The authors demonstrated through several experiments that DD-NHL is superior to NHL should historical data be available. However, a more exhaustive study [104] concluded that even this variant yields poor performance in classification scenarios.

3.2 Error-driven approaches

Error-driven learning methods aim at generating weight matrices that minimize an error function based on the difference between the expected responses and the map-inferred outputs. These algorithms are more expensive optimization techniques given that they attempt to fit a model to a set of historical observations. On the other hand, they require the definition of the objective function to be optimized, which is the core of the learning procedure. In this subsection, we revise some of these objective functions and their semantics.

Koulouriotis et al. [54] applied a genetic strategy to learn the model structure from data. In this approach, the method uses a collection of input/output vector pairs, which are referred to as examples. The method computes a weight set that transforms the input vectors into the output vectors. Equation (15) shows the error function to be minimized, where x encodes the weight set, K is the number of instances, M is the number of concepts, while A_{ki} and $\tilde{A}_{ki}$ denote the current response and the expected one, respectively.

$$E(x) = \sum_{k=1}^{K} \sum_{i=1}^{M} |A_{ki} - \tilde{A}_{ki}| \tag{15}$$

Anologously, a learning procedure based on Particle Swarm Optimization (PSO) was introduced in [105]. Similarly to the above approach, the algorithm computes the weight set on the basis of historical data (multiple sequences of state vectors) that converge into a desired final state. Therefore, the model requires human knowledge to specify adequate constraints that would guarantee that the relationships within the FCM model retain the physical meaning defined by the experts. Equation (16) unveils the error function to be minimized for the k-th instance, where A^*_{ik} denotes the activation value of the i-th decision concept, $H(x)$ is the well-known Heaviside function [87], whereas L_i and U_i represent the lower and upper boundaries, respectively, of the acceptable activation interval.

$$E_k(x) = \sum_{i=1}^{M} H\left(L_i - A^*_{ik}\right) |L_i - A^*_{ik}| + \sum_{i=1}^{M} H\left(A^*_{ik} - U_i\right) |A^*_{ik} - U_i| \tag{16}$$

A memetic approach that combines PSO with both deterministic and stochastic local search schemes for solving the above problem was investigated in [110] [111]. The memetic schemes were applied to real-life problems and compared with well-known continuous optimizers, thus justifying their superiority.

In [89], the authors adopted Differential Evolution (DE) to compute the weight matrix by using a single input-output record. Equation (17) shows the error function to be minimized in this learning procedure.

$$E(x) = \sum_{i=1}^{M} |L_i - A^*_{ik}| + |A^*_{ik} - U_i| \tag{17}$$

Another approach for learning the network structure from a single historical record was proposed by Mateou et al. [64] where the weights are recalculated to increase the reliability of FCM-based systems in multiobjective decision-making

scenarios. The authors used a variant of Genetic Algorithms (GAs) for multiobjective optimization that attempts estimating a weight matrix while satisfying several criteria at the same time. Equation (18) displays the error function for this model, where $\tilde{A}_i$ is the expected response for the i-th concept, $A_i^{(t)}$ is the current activation value and $T > 50$ denotes the maximal number of time steps. It should be stated that we have rewritten the original formulation of this equation to be consistent with the notation used in this paper.

$$E(x) = \sum_{i=1}^{M} |\tilde{A}_i - \sum_{t=1}^{50} \frac{A_i^{(T-t)}}{50}| \tag{18}$$

Stach et al. [124] [125] proposed a genetic learning method that only requires a single activation sequence. Equation (19) displays the attached error function, where $p \in \{1, 2, \infty\}$ and c denotes the normalization factor, i.e., $c = 1/(T-1)M$ for $p \in \{1, 2\}$ and $c = 1/(T-1)$ for $p = \infty$. This procedure decomposes the input sequence into at most $K = T - 1$ pairs of the form $\{A^{(t)}, A^{(t+1)}\}$ such that $A^{(t)}$ defines an initial state vector and $A^{(t+1)}$ is the expected response. Repeated pairs are not considered since if the recurrent system reaches a previously produced state, its behavior will be exactly the same regardless of the simulation history. Consequently, all state vectors that are produced after either a limit cycle or an equilibrium point is reached are already ignored.

$$E(x) = c \sum_{t=1}^{T-1} \sum_{i=1}^{M} |A_i^{(t)} - \tilde{A}_i^{(t)}|^p \tag{19}$$

The above error function is perhaps the most widely used and accepted in the context of FCM learning. However, its performance quickly deteriorates as the number of concepts increases. In order to improve the algorithm scalability, Stach and his colleagues [121] examined the parallel nature of Real-coded Genetic Algorithms (RCGA). The proposed master-slave parallelization scheme allows learning FCM-based model comprised of dozens of concepts.

Later on, a divide-and-conquer procedure on the basis of RCGA [123] was put forth. This technique makes use of a strategy to subdivide the input data in order to speed up the learning process. Equation (20) displays the fitness function to be maximized, where α and β are the positive scaling constants.

$$F(x) = \frac{\alpha}{\beta \sum_{t=1}^{T-1} \sum_{i=1}^{M} \left(A_i^{(t)} - \tilde{A}_i^{(t)}\right)^2 + 1} \tag{20}$$

Chen et al. [18] adopted an error function that uses multiple input sequences to improve the generalization capability of the learned model. Equation (21) shows this function, where K denotes the number of training sequences and T the number of time steps. The reader may notice that this error function is a generalization of Equation (19) in which a single input sequence is required. In order to accomplish the error function minimization, the authors decomposed each numerical weight into a sequence of $P + 1$ discrete variables, i.e., the sign and P integer variables denoting the required precision. The algorithm was able to successfully tune a fully connected network topology comprised of 40 concepts.

$$E(x) = \frac{1}{KM(T-1)} \sum_{k=1}^{K} \sum_{i=1}^{M} \sum_{t=1}^{T} \left(A_{ki}^{(t)} - \tilde{A}_{ki}^{(t)} \right)^2 \tag{21}$$

Subsequently, Chen et al. [19] introduced a decomposed learning scheme based on Swarm Intelligence to adjust gene regulatory networks comprised of 100 nodes. Later, Chen and their collaborators [20] proposed a modified error function that includes a sparseness penalty factor p_s as depicted in Equation (21). In this case, they performed the optimization process using a decomposed RCGA. The experiments demonstrated that this algorithm was able to learn large-scale networks of up to 300 concepts. To the best of our knowledge, this method stands as the best published result to date in learning complex FCM networks.

$$E(x) = \frac{1}{KM(T-1)} \sum_{k=1}^{K} \sum_{i=1}^{M} \sum_{t=1}^{T} \left(A_{ki}^{(t)} - \tilde{A}_{ki}^{(t)} \right)^2 + p_s \sum_{i=1}^{M} \sum_{j=1}^{M} |w_{ij}| \tag{22}$$

Several heuristic search methods have been used to optimize the above functions. For example, Alizadeh et al. [8] adopted Tabu Search while Ghazanfari et al. [31] resorted to the search capabilities of Simulated Annealing and Genetic Algorithms. Other search procedures include: Artificial Inmune Systems [59], Chaotic Simulated Annealing [6], Big Bang - Big Crunch [136], Extended Great Deluge [12], Artificial Bee Colony [135], Particle Swarm Optimization [82], Cultural Algorithm [4], Bacterial Evolutionary Algorithm [16], Structure Optimization Genetic Algorithm [112], Imperialist Competitive Algorithm [3], etc.

It is opportune to highlight that some review papers (e.g., [87] [97]) use the term *population-based* to gather both single-trajectory and population-based search methods, leading to a theoretically inaccurate taxonomy. This (almost imperceptible) mistake comes from the fact that the first supervised learning algorithms were based on population-based algorithms, which often produce better results. On the other hand, we cannot claim the novelty of a learning method only because we adopted a different optimizer, even when we understand that this component is a key piece in most data-driven learning techniques.

3.3 Two different approaches - what works when?

As mentioned, the majority of the FCM learning algorithms are devoted to computing a "suitable" weight set. Nevertheless, each kind of learning algorithm comes with its own set of advantages and limitations, which makes it appropriate to specific types of problems depending on the data and knowledge availability. Selecting an adequate approach in each case is not trivial and may depend on obtaining an accurate and theoretically sound modeling.

Hebbian-based methods are convenient to fine-tune the weight set with a small deviation from the initial configuration provided by the expert. This means that the adjusted weights might preserve their causal meaning, which cannot be ensured when using a data-driven algorithm. However, their poor generalization capability arises as a barrier difficult to overcome. This issue render Hebbian-based methods unfit prediction problems comprising two or more categories, unless we use near-optimal initial weights and multiple training examples.

As an alternative, we can use error-driven learning algorithms. Several studies have shown that these techniques increase the FCM functionality, robustness and generalization abilities [20] [87]. This suggests that a single weight set is capable of recognizing patterns belonging to different decision classes. In spite of these remarkable advantages, error-driven algorithms also have relevant drawbacks to be considered. For example, they are time-consuming and require multiple input-output training sequences to compute the learned model.

Perhaps the chief downside of error-driven learning methods is that they provide solutions that are difficult or impossible to interpret, which may lead to incorrect static analysis [98]. In a nutshell, we cannot guarantee that the algorithm is capable of producing authentic causal relations corresponding to the behavior of the physical system under investigation. Furthermore, these algorithms might generate FCM models with poor convergence features.

From the above analysis we can conclude that Hebbian-based methods might be adequate to face control problems [116] where the domain restrictions are clearly known. In such a case, these unsupervised procedures will adjust the initial weight matrix according to an activation sequence. Notice that the Oja's learning rule generates an algorithm for principal components analysis rather than an accurate predictive model; perhaps that is why FCM-based classifiers perform poorly when trained with Hebbian-based algorithms. In contrast, error-driven approaches are more convenient for solving pattern classification and forecasting problems where multiple training sequences are available. However, in these scenarios the network interpretability may be compromised since there is no guarantee that the produced weight matrix encloses authentic causal relations.

3.4 Hybrid learning approaches

Hybrid learning approaches employ a combination of the first two mentioned FCM learning mechanisms: Hebbian-based and error-driven procedures. In the hybrid learning schemes, the learning goal is to modify/update the weight matrices on the basis of initial knowledge from the experts and historical data in a two-stage process [87]. The literature is rather scant when it comes to describing hybrid FCM learning methods and, for some reason, the proposed approaches are not widely accepted in solving real-world FCM-based problems.

Papageorgiou and Groumpos [90] investigated a coupling of the DE and NHL algorithms by using both the global search capabilities of evolutionary algorithms and the effectiveness of the NHL rule. This hybrid learning technique was successfully applied to real-world decision making problems. The experimental analysis results confirmed that this hybrid strategy was capable of effectively training FCMs, thus leading the system to desired states and determining an appropriate weight matrix for each specific problem.

Another hybrid scheme was presented by Yanchun and Wei [133]; the authors put together an RCGA-based optimization scheme and the NHL algorithm to solve a partner selection problem. Their algorithm inherited the main features of each learning technique, the RCGA population-based algorithm and the NHL rule, thus combining expert knowledge with available data.

A hybrid FCM learning procedure blending NHL and the Extended Great Deluge Algorithm (EGDA) was investigated in [113]. This hybrid scheme possesses NHL's efficiency and EGDA's global optimization capabilities. The FCM is first trained via NHL in order to get a set of weights close to the optimal (unknown) topology, and then EGDA further optimizes the network structure driven by the response error minimization. The reported results are really encouraging but deeper research using more complex scenarios is required.

3.5 Other learning approaches

During the literature review, we identified other alternative learning schemes that do not fall within the three aforementioned groups. Such methods combine other theories to build and/or optimize an FCM-based system.

Konar and Chakraborty [47] proposed a model for unsupervised learning and reasoning on a special type of cognitive maps realized via Petri nets. The learning process in this context adapted the weights of the directed edges from transitions to places in the Petri net. After convergence of the learning algorithm, the network can be used to compute the beliefs of the desired propositions from the supplied beliefs of the axioms. Later, Kyriakarakos et al. [56] merged FCM and Petri nets for autonomous polygeneration microgrids.

Carvalho and Tomé [17] introduced an approach based on fuzzy Boolean nets for learning rule-based FCMs such that fuzzy Boolean nets are used to extract the linguistic membership functions from historical data. The internal binary memories of each consequent concept are modified according to the states of the antecedent concepts and the state of that consequent entity.

Madeiro and Zuben [63] introduced a gradient-based method for the automatic construction of FCM-based systems. The objective function is minimized starting from an initial solution given that gradient-based methods are local optimization algorithms. This technique leaned on RCGA with gradient search to learn the FCM weights and construct the network structure. It should be mentioned that gradient-based leaning methods are also error-driven approaches, but they do not use a metaheuristic to minimize the least-squares error function. That is why we revise gradient-based methods in this section separately.

A supervised learning procedure based on the gradient method was proposed by Gregor and Groumpos [34] to compute the weight set in the direction of the steepest descent of the error function. As well, a multi-step gradient method [95] was applied to the domain of time series forecasting, more explicitly, to predict the electricity consumption and stock exchange returns. Although this gradient-based method seems like a promising approach for FCM construction using time series data, further analytical experimentation is required.

The evolutionary mechanism of *Cellular Automata* was exploited in [21] to learn the FCM connection matrix. A one-dimensional cellular automata encoded the weight set and the cellular states were chosen within the $[0, 1]$ range to create a cell space. In order to guide the optimization effectively and accelerate the convergence speed, a mutation operator was added to the algorithm. This approach was tested in short-term stock prediction.

Luo et al. [62] proposed an FCM model to design game-based learning systems inspired on their properties for concept representation and reasoning. It computes the difference between the outputs of the teacher and the learner submodels to control the whole game learning process. The improved FCM has the ability to self-learn from both existing data and a priori knowledge.

We should acknowledge the existence of previous studies on this topic, for instance Stach et al.'s survey [120] that goes over the main features of the Hebbian-type and evolutionary-based learning algorithms for FCMs. Likewise, the survey paper in [87] gives continuity to this important field.

3.5.1 Network topology optimization

Several learning algorithms have been developed to optimize the topology of complex causal networks. For instance, Alizadeh et al. [7] put forth an FCM learning method that simplifies the topology by only selecting relevant concepts without sacrificing the model's accuracy. In this method, the FCM concepts were clustered on the basis of their cause and effect behaviors.

Nápoles et al. [74] designed a learning algorithm to produce compact FCM-based systems. In the first step, the algorithm automatically estimates the causal weight matrix from historical data. Next, the superfluous/irrelevant concepts are removed by minimizing a constrained error function that ensures preserving the overall accuracy. This algorithm uses the centrality of each concept as a metric to guide the search across a space comprised of 2^M solutions.

Similarly, Homenda et al. [40] investigated the effect of the *a posteriori* removal of weak connections or concepts. The concepts were removed together with their connections. After modifying the network topology, the simplified models were tested with well-known time series to check whether they were still able to predict future states with an acceptable low error or not.

A new Structure Optimization Genetic Algorithm (SOGA) for FCM learning came to light in [112] for modeling complex decision support systems. The method defines an error function with an additional penalty for those FCM topologies with large number of concepts and connections among them. The obtained results confirmed that SOGA was able to reduce the FCM structure while preserving its most important concepts and connections.

In [92] the authors proposed a reduction approach for FCMs through a clustering algorithm based on fuzzy tolerance relations. The algorithm finds concepts that have identical or similar behavior and groups them into the same cluster. Next, the causal relations are recalculated and parameters associated with the sigmoid transfer function are adjusted to compensate for these modifications. Notice that merging concepts lead to the creation of artificial concepts and relations that may not have a clear meaning for the problem at hand.

More recently, Salmeron and Froelich [115] introduced a dynamic optimization method for FCMs in time series forecasting. The algorithm estimates the weight matrix, the set of relevant concepts and the transfer function (together with its parameters) in a single step. This approach allows optimizing the network structure during the learning stage, thus avoiding the creation of semantically meaningless concepts. We strongly recommend using this methodology, as many FCM scenarios can be successfully mapped to time series problems.

3.5.2 Optimization of convergence features

As mentioned in Section 6, Nápoles et al. [69] [70] [80] [71] proposed several models to improve the convergence of FCM-based systems. In these methods, the authors assume that the concepts and causal relations have been previously estimated. To accomplish the learning goal, the proposed algorithm estimates the parameters of the sigmoid functions attached to the map concepts. In spite of the promising results obtained in the classification field, the study conducted in [71] evidences that improving the convergence features of FCM-based systems is not always possible without altering the semantics behind the cognitive model.

4 Fuzzy Cognitive Maps for Time Series Forecasting

Fuzzy cognitive mapping has been widely used for time series forecasting. In this section we elaborate on this particular application due to its importance in data stream mining and the modeling of dynamic systems.

Let $y \in \Re$ be a real-valued variable observed over a discrete time scale within the period $t \in [1, 2, \ldots, N]$, where $N \in \mathbb{N}$ denotes its length. Thus, a univariate time series is defined as a sequence of observations $\{y^{(t)}\} = \{y^{(1)}, y^{(2)}, \ldots, y^{(N)}\}$. Let us denote by *historical* time series the one in the period $t \in [1, 2, \ldots, t_e]$, where $t_e \leq N$. The goal of forecasting is to predict the next values of the time series, i.e., $\hat{y}^{(t_e+1)}, \hat{y}^{(t_e+2)}, \ldots, \hat{y}^{(t_e+H)}$, where H is referred to as the *prediction horizon*. To accomplish the forecasting task, a model $\mathcal{F}$ is required. Assuming single-step forecasting, i.e., for $H = 1$, the model $\mathcal{F}$ is used to calculate $\hat{y}^{(t_e+1)} = \mathcal{F}(y(t_e))$. The problem to be addressed is the construction of $\mathcal{F}$, which is usually not known and has to be discovered using historical records.

At every time (i.e., iteration) step within the prediction period $t \in [t_e + 1, N]$, an individual forecasting error is evaluated as $E^{(t)} = \hat{y}^{(t)} - y^{(t)}$. To evaluate the cumulative prediction error over the entire period $[t_e + 1, N]$ several different benchmarking measures may be adopted. Two of the most popular ones are the *Mean Absolute Percentage Error* (MAPE, see Equation (23)) and the *Root Mean Squared Error* (RMSE, see Equation (24)). The lower the value of these measures, the more precise the forecasting model becomes.

$$MAPE = \frac{1}{N} \cdot \sum_{t=1}^{N} \left| \frac{E^{(t)}}{y^{(t)}} \right| \cdot 100\% \tag{23}$$

$$RMSE = \sqrt{\frac{1}{N} \cdot \sum_{t=1}^{N} \left[\frac{E^{(t)}}{y^{(t)}} \right]^2} \tag{24}$$

In the FCM context, the forecasting model $\mathcal{F}$ corresponds to the FCM-based network. This means that the FCM concepts are mapped to time series variables and then, the learning of the FCM weights occurs followed by the testing within the prediction horizon. Depending on the type of mapping between the FCM concepts and the lagged values of the time series, the FCM can be used to forecast both univariate and multivariate time series. In addition, FCMs have been adapted to the forecasting of approximated time series, interval-valued time series or granular time series, as it will be illustrated in the following subsections.

4.1 Univariate time series forecasting

Earlier FCM applications to time series forecasting were confined to the univariate case. In these scenarios, the FCM concepts are mapped to the lagged time series, thus playing a role similar to that of the regressors in auto-regressive forecasting models. Such an approach was proposed in [38] [41] [60] [61].

In these works, all values of the learning part of the time series are clustered through the fuzzy c-means method, where the number of clusters c is a user-defined parameter. In this way, at each time step, the value of the time series belongs to every created cluster with some membership grade. It is assumed that every cluster (denoted as an FCM concept) plays the role of a fuzzy set. This suggests that the activation value A_i is the degree to which the current value of the time series $y^{(t)}$ belongs to the concept C_i. By performing this operation for all concepts, the vector of concept activation states is obtained, while the FCM weights are learned using the training part of the data. During the prediction phase, the sequence of activation vectors is reconstructed using the learned model. To obtain numerical values of the forecasted time series, each vector is defuzzified.

In [41] an FCM simplification was performed. The edges having weak absolute value of weights were pruned from the FCM topology. Different values of pruning thresholds were examined with regards to their influence on the obtained prediction errors. It turned out that around 1/6 of the edges could be dropped without a substantial increase in the prediction error. A similar approach was used in [60] where a higher-order FCM was employed as the forecasting model. This means that the FCM's reasoning equation was modified to include not only the values of $A_j^{(t-1)}$ but also older states of concepts $A_j^{(t-2)}, A_j^{(t-3)}, \ldots, A_j^{(t-k)}$.

Another approach related to the times series forecasting using FCMs was proposed in [39] [115]. It assumes a direct mapping of the lagged values of the time series to the FCM concepts. The fuzzification function was replaced with the max-min normalization. Learning and testing was performed assuming online availability of data. The concept of moving (sliding) window was applied for learning and testing purposes. In addition to the previous works, the learning of the FCM proposed in [115] encompassed the optimization of the length of the learning period and the selection of the transformation function together with its parameters. The optimization was performed dynamically for every learn-and-test trial. Empirical results suggested that the obtained FCM model is very competitive with respect to other models in terms of the prediction accuracy.

4.2 Multivariate time series forecasting

In the case of multivariate time series, the vector $Y = [Y^{(1)}, \ldots, Y^{(t)}, \ldots, Y^{(N)}]$ comprises a sequence of observations for multiple real-valued variables, such that $Y^{(t)} = [y_1^{(t)}, \ldots, y_i^{(t)}, \ldots, y_M^{(t)}]$. Therefore, the single-step forecasting assumes the form: $\hat{Y}^{(t_e+1)} = \mathcal{F}(Y^{(t_e)})$, where the network $\mathcal{F}$ is used as the forecasting model. It is possible to distinguish two approaches to the forecasting problem. In the first case, it is assumed that the multivariate time series $Y^{(t)}$ is synthetic and generated by the existing cognitive network. This implies that the learning phase will be oriented to compute a model that mimics the generated time series. Additionally, the reconstruction of the original FCM should be considered.

In [126] the authors adopted this first approach and proposed a new method for time series prediction, which was carried out both at the linguistic and numerical levels. The proposed prediction method combines FCMs with granular, fuzzy-set-based input models. The obtained results, which are compared with other prediction models using fuzzy sets, have shown that the proposed architecture achieves good accuracy expressed at both the linguistic and numerical levels.

The other approach revolves around the application of FCMs to the forecasting of real-world time series. Froelich and Juszczuk [26] elaborated on the predictive capabilities of adaptive and evolutionary FCMs. The goal of their research was to determine what learning methods should be recommended for a particular prediction problem. They illustrated the FCM predictive capabilities through an example concerning the forecasting of weather conditions.

Froelich et al. [27] put forth an FCM-based prostate cancer prediction model. Moved by the problem domain requirements, an improved evolutionary approach for learning the FCM model was designed. The focal point of the new method was to improve the effectiveness of long-term prediction. The evolutionary approach was experimentally verified using clinical data acquired during a two-year period. The advantage of using this method was theoretically justified and then empirically corroborated in [88] with the introduction of a multi-step approach to learning evolutionary FCMs for the prediction of pulmonary infection.

Song et al. [118] [119] designed an FCM-based network using neural networks and fuzzy sets in order to predict chaotic time series. The four-layer fuzzy neural network aimed to enhance the FCM's learning ability and coupled the inference mechanism of conventional FCMs with the learning of fuzzy membership functions. In this scheme, mutual subsethood is used to define and describe the causalities in the cognitive model. As a further advantage, the FCM model can be automatically constructed from data, and therefore the expert's intervention is not required. Simulation results confirmed that this approach performs better in terms of both prediction accuracy and architectural simplicity in most cases, compared to the methods adopted for benchmarking purposes.

Recently, Vanhoenshoven et al. [131] use an approach based on autoregressive integrated moving average to overcome the convergence issues of the recurrent network while preserving its capability of performing multiple-ahead predictions. Not too many FCM-based forecasting models can claim this feature. The preliminary results are certainly encouraging, but further research is required. Similarly, Alghzawi [5] illustrated the negative effects of the convergence on the FCM-based forecasting models by using a real-world problem concerning social security revenues in Jordan. In reference [109] the reader may find other FCM applications to the forecasting of real-world multivariate time series.

4.3 Approximate time series forecasting

It is worth mentioning that some applications do not require to deal with accurate numerical time series. Instead, only approximating the numerical expected outputs is desired. For example, in meteorology, often minimal and maximal daily temperatures are important. The exact variability of temperature during the daytime is usually not required. For this reason, the original time series is approximated and only this approximation is subject to forecasting.

Froelich and Salmeron [29] approximated the original time series through intervals, where the lower and upper bounds of the intervals made up the interval time series (ITS). This multivariate ITS model was successfully validated using real-world meteorological data. Moreover, aiming at handling the multivariate ITS, the FCM was appropriately adapted by replacing classical numerical operators with those from interval arithmetic. The resulting fuzzy grey cognitive map (FGCM) [114] was deemed as the forecasting model. To cope with interval arithmetic in the FGCM, a modified genetic algorithm was developed.

Another model to the approximation and forecasting of time series was introduced in [28]. A numeric time series was approximated via information granules in the form of triangular fuzzy numbers. The sequence of these granules constituted a granular time series (GTS). For the GTS forecasting, a specialized model had to be learned. The model applied clustering of fuzzy numbers and a genetic algorithm to compute the parameters related to forecasting scheme.

5 Fuzzy Cognitive Maps for Classification

As mentioned, FCMs have been widely studied due to their appealing properties for handling complex and dynamic systems, but less attention has been paid to the development of FCM-based classifiers.

The *pattern classification* problem [24] is about building a mapping $f : \mathcal{U} \rightarrow \mathcal{D}$ that assigns to each instance $x \in \mathcal{U}$ described by the attribute set $\Phi = \{\phi_1, \ldots, \phi_M\}$ a decision class D from the N possible ones in $\mathcal{D} = \{D_1, \ldots, D_N\}$. The mapping is often learned in a *supervised* fashion, i.e., by relying on an existing set of previously labeled examples used to train the model. The learning process is regularly driven by the minimization of a cost/error function.

Machine Learning researchers are mainly focused on attaining high prediction rates. Some classifiers like *Artificial Neural Networks*, *Support Vector Machines*, *Ensemble techniques* or *Random Forests* are well known to be the most likely successful algorithms for addressing a real-world problem in terms of prediction rates. Regrettably, most accurate classification models do not provide any mechanism to explain how they arrived at a particular conclusion and behave like a "black-box". This negatively affects their practical usability in scenarios where an understanding of the inference process is required.

5.1 Low-level Fuzzy Cognitive Classifiers

The first attempt to use FCMs in solving classification tasks was implemented in [101] [100]. In these papers, the authors defined the notion of *FCM-based classifiers*. On of the challenges to be faced when constructing an FCM-based classifier lies in how to connect input and output concepts since an FCM classifier's topology (i.e., concepts and causal relations) must comprise a coherent meaning for the system being modeled. If the input concepts denote features of the classification problem, then we are in presence of a *low-level cognitive classifier* where neural processing units can be categorized as shown below:

Definition 1 We say that a concept C_i is an *independent input concept* if its activation value does not depend on any other input concept.

Definition 2 We say that a concept C_i is a *dependent input concept* if its activation value is influenced by other connected concepts.

Definition 3 We say that a neural processing entity C_i in an FCM-based classifier is an *output concept* if we can predict a decision class from its final activation value, which only depends on the connected input concepts.

Typically, independent and dependent input concepts are used to activate the cognitive networks since they often denote problem features. Output concepts, on the other hand, are used to compute the decision class for an initial activation vector. In the case of independent input concepts, they can propagate the initial activation vector and they are not influenced by other input concepts, therefore their activation value remains static. Notice that the domain expert must ideally determine the role of each concept and the way that input concepts are connected among themselves. In spite of this fact, Papakostas et al. [104] proposed three generic architectures for mapping the decision classes:

- **Class-per-output architecture**. Each decision class is mapped to an output concept. Therefore, the predicted decision class corresponds to the label of the output concept having the highest activation value.
- **Single-output architecture**. Each decision class is enclosed into the activation space of a single output concept.
  1. *Using a clustering approach.* Each class is associated with a cluster center. In the testing phase, the center having the closest distance to the projected activation value is assigned to the input instance.
  2. *Using a thresholding approach.* Each decision class is associated with a pair of thresholds. In the testing phase, the interval comprising the projected activation value is then assigned to the input instance.

In these architectures, the experts should ideally determine the way in which concepts are connected while the weights are often sought via a learning method. This implies that the human intervention is required during the construction stage. Even so, automatic construction methods based on metaheuristics are unable to produce authentic causal relations since they just fit the model to the existing data. Therefore, we are losing the interpretability features attached to the network, although the decision process remains transparent.

On the other hand, the absence of hidden processing entities in these recurrent neural networks may probably lead to poor prediction rates. Aiming at boosting the prediction capability of FCM-based classifier, in [100] the authors presented two hybrid topologies. Figures 2 and 3 show these topologies that include a *black-box* classifier to improve the prediction rates.

In the first hybrid model, the black box produces a confidence degree per decision class. This confidence vector is used as the initial configuration for the FCM model that corrects the outputs produced by the black box. In the second model, the input concepts are also connected to the output ones, so the predictions computed by the black-box classifier can be understood as a bias. However, these classification models greatly reduce (or perhaps completely suppress) the transparency attached to the cognitive network. If this happens, then there is no reason to use FCMs in classification scenarios; instead, we can adopt powerful black boxes like Support Vector Machines or Random Forests.

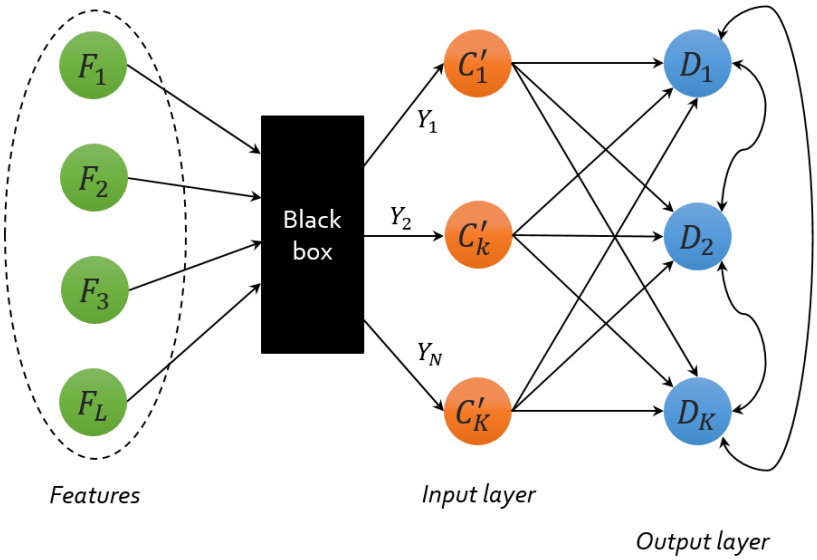


Fig. 2: Hybrid FCM-based classifier type-1.

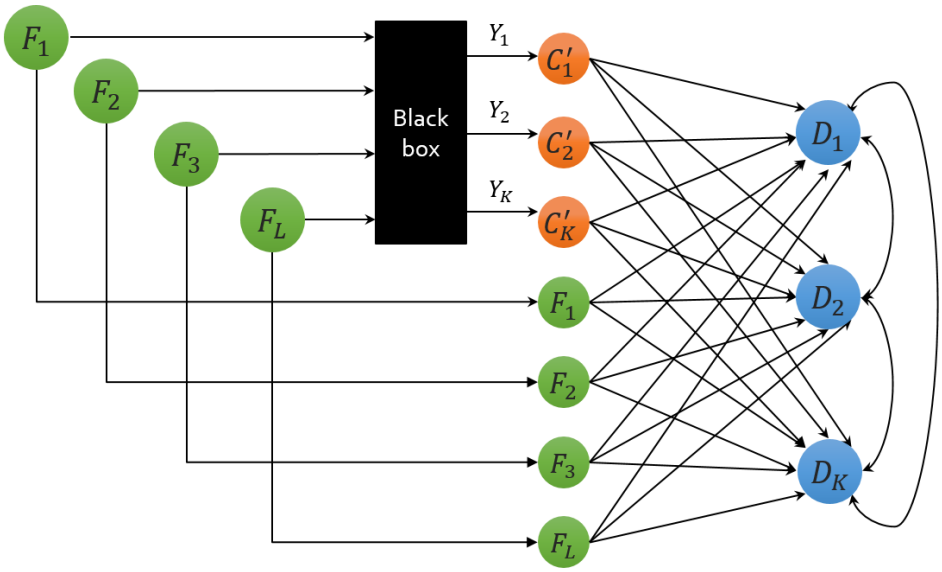


Fig. 3: Hybrid FCM-based classifier type-2.

In the FCM literature, several studies [84] [94] [93] [83] [117] using Hebbian-based algorithms in pattern classification scenarios are reported. However, some of these works are far from being considered authentic pattern classification solutions. Papakostas et al. [104] thoroughly tested the performance of several Hebbian-type algorithms in classification scenarios and concluded that these learning procedures regularly produce very poor classification rates. For example, the NHL algorithm reported an average performance of 15.47% on the *Glass* dataset whereas DD-NHL computed an average prediction rate of only 4.91%! Of course, we could find specific pattern classification scenarios on which a Hebbian-based algorithm performs comparably to other learning approaches.

In the supervised learning case, the authors in [82] resorted to a Swarm Intelligence method for predicting autistic disorder in children. The network topology was initially proposed in [45] and comprises 24 concepts: 11 independent input concepts, 12 dependent input concepts and a single output concept using a thresolding approach. Afterwards, Kannappan and Papageorgiou [44] addressed the same classification problem by means of artificial immune systems.

Zhou and Zhang [137] came up with a text categorization method on the basis of similarity-based rough sets and FCMs, which replaces the FCM causal relation with a correlation relation to generate text classifiers.

An FCM model for the screening and separation of usual ductal hyperplasia from the rest of intraductal lesions was described in [9]. For this goal, 86 patients in the Shahid Beheshti Hospital of Isfahan were studied, and 10 of the most significant features were extracted by three pathologists and turned into FCM concepts when determining the screening of these lesions. An overall classification accuracy of 95.35% was obtained on the entire database.

Nápoles et al. [74] proposed an FCM-based classification method using a single-output architecture and a thresholding approach to model HIV proteins. In this network, each sequence position is denoted as an independent input neuron due to the fact that a single-point mutation may certainly encourage changes in other positions. The goal is to predict whether a new mutation sequence is resistant to the target inhibitor or not. The prediction rates achieved by this model ranged from 95% to 99% depending on the target inhibitor. In an attempt to understand the virus behavior, Grau and Nápoles [32] employed these fine-tuned networks to synthetically generate mutations having susceptible features.

Froelich [25] introduced a new algorithm to improve the prediction rates of FCM-based classifiers using a single-output architecture. The algorithm generates thresholds with a superior discriminatory power, which are determined after learning the FCM structure. The results of the experiments conducted using publicly available Machine Learning datasets evidenced that the application of the proposed algorithm leads to an improved accuracy of the FCM classifier.

5.2 Granular Fuzzy Cognitive Classifiers

Granular FCM classifiers refer to high-level fuzzy cognitive models where input concepts map information granules [13] rather than low-level features. For example, Nápoles et al. [78] [77] introduced the notion of *rough cognitive mapping* in the context of pattern classification. The new classification model transforms the

original feature space into a granular one that is exploited using the FCM inference rule. In these so-called *Rough Cognitive Networks* (RCNs), the weight matrix is automatically computed on the basis of the three-way decision rules [134] from Rough Set Theory (RST) [106] [1] that define three regions to perform the classification process. The RCN model achieved competitive performance with respect to state-of-the-art classifiers in traditional classification problems [78] [77] as well as a network intrusion detection scenario [75].

Recently, two RCN models were introduced: *Rough Cognitive Ensembles* [72] and *Fuzzy-Rough Cognitive Networks* [79]. The purpose of these methods is to deal with the RCN's parametric requirement while preserving their global prediction capabilities. Although both variants performed comparably, it should be remarked that we can achieve the same accuracy using an ensemble composed of a few networks that using a single fuzzy-rough classifier! Furthermore, the fuzzy-rough classifier allows smoothly elucidating its decision process based on its foundations: fuzzy inclusion degrees and causal relations among granules.

Inspired on the RCN semantics and the approaches discussed in [107] and [108], Nápoles et al. [73] proposed a *partitive granular cognitive map* to solve graded multi-label classification problems. In these machine learning problems, the goal is to predict the degree to which each instance relates to each available decision class. Three different FCM topologies were studied and several convergence features were included into the supervised learning methodology. Numerical experiments confirmed the ability of these granular classifiers to accurately estimate the degree of association between an object and each label.

The above references are not the only works in this realm but they serve to demonstrate the rapid growth and diversity of application areas that have witnessed the benefits of FCM-based solutions.

6 Software Tools for Fuzzy Cognitive Maps

While reviewing the literature, we found some attempts to develop software tools for creating and experimenting with FCMs. Papers related to this methodology usually present theoretical methods or practical applications, but they are rarely supported by well-defined software implementations. In this section, we review the most relevant software available in the literature.

FCM Modeler [68] is a pioneering software for designing FCM-based systems. It was developed about 20 years ago and consists of a simple interface with the aim of supporting group decision making on a qualitative static model. Some of the features of this software tool include: (i) intuitive user interface, (ii) the capability of designing and storing FCM-based systems, and (iii) the inference of FCM models based on the observed concepts and successive state vectors. FCM Modeler was intended to serve as a general modeling tool but the project never evolved in that direction. The authors also advocate for the inclusion of a basic implementation of a machine learning algorithm. We could not find other references in the literature referring to this tool; nevertheless, we acknowledge it as a seminal idea that seeded other ensuing implementation endeavors.

A very similar approach termed *FCM Designer* [2] shows a much better implementation and extends the original idea behind FCM Modeler. This software tool is a notch higher than the previous one but it is still rather difficult to interact with. FCM Designer's chief functionalities include: (i) interactive graph visualization, (ii) graphical support to design FCM models and (iii) the ability to simulate new scenarios using the available causal knowledge. Likewise, this software tool allows choosing the inference rule by picking the type of transfer function and the stop criterion. The key drawback of this software lies on the lack of learning algorithms to compute the parameters that define the system.

Mental Modeler [33] is another recently proposed software tool. It features a web-based modeling implementation to support group decision making, thus allowing experts to collaboratively represent and test their assumptions about a system. Mental Modeler can be mainly used by non-IT people, usually experts or stakeholders in a given domain who need to design a simple cognitive map (with signed and weighted relationships) and simulate its behavior for some scenarios. The most important disadvantages of this tool are related to the lack of learning algorithms and its limited set of experimental options. However, the web-based approach is appreciated, especially to be used by unsavvy users in the field.

The *Java Fuzzy Cognitive Maps* (JFCM) emerged as an open-source library for FCM modeling [22] in 2014. The library is small and simple but can be used to create a variety of FCM-based models. JFCM allows loading networks directly from XML files, thus increasing its usability. The central idea behind this project is to create reusable modules that could be loaded when seeking an FCM-based solution to a given problem. JFCM was conceived from an object-oriented programming perspective; hence, if the set of standard components are not sufficient for more complex projects or theoretical proposals, the source code may be extended accordingly. This advantage becomes a limitation for experts with no programming skills since it requires a straight plunge into the source code.

Intelligent Expert System based on Cognitive Maps (ISEMK) is a software for modeling decision support systems based on fuzzy cognitive maps and artificial neural networks [112] [96]. ISEMK is composed of four basic blocks, namely: the knowledge processing, analysis of the FCM operation, neural network tool and graphical user interface. The first block contains an FCM module and learning algorithms based on gradient method with the use of historical data and population-based learning via RCGA and SOGA as optimizers. The neural network module allows implementing a multi-layer neural network for time series forecasting as well as two learning algorithms: the Levenberg-Marquardt method [35] and the Backpropagation method with momentum [36]. Moreover, ISEMK has an interface supporting the visualization of results.

FCM Tool was originally introduced by León et al. [57] to model a real decision-making problem concerning public transportation in Belgium. This software allows (i) designing complex FCM-based models through an interactive graph visualization, (ii) customizing the update rule by selecting the kind of transfer function and stop criterion to be used, (iii) analyzing scenarios and their effects over the whole system. FCM Tool provides a population-based learning algorithm for automatically deriving causal weights from historical data. Another relevant feature is the inclusion of aggregation operators for combining several FCM-based systems into a single knowledge-based representation.

FCM Tool eventually evolved into *FCM Expert* [76], a general-purpose and more complete software platform for modeling FCM-based systems. As mentioned, FCM Tool was meant to address a specific decision-making problem. This implies that the implemented learning algorithms could not be used to tackle more general pattern classification problems. FCM Expert inherits the strongest features in FCM Tool and adds several unsupervised and supervised learning algorithms for adjusting the weight matrix. This software tool includes techniques for optimizing the network topology [74] and improving the system convergence [70] [80] without losing relevant information. Moreover, the user can configure the model parameters (e.g., the transfer function, the reasoning rule or the stop criterion) and benefit from a friendly graphical interface.

Table 1 provides a comparison of the reviewed software tools across several often desired indicators such as the existence of experimentation facilities, inclusion of machine learning algorithms and graphical support.

Table 1: Comparison of existing software tools for FCM modeling

	Year	Simulation options	Learning algorithms	Graphical support
FCM Modeler	1997	None	Only one	Poor
FCM Designer	2005	Limited	None	Adequate
FCM Tool	2011	Several	Only one	Advanced
JFCM	2013	For developers	None	None
Mental Modeler	2013	Limited	None	Adequate
ISEMK	2015	Several	Several	Adequate
FCM Expert	2017	Several	Several	Advanced

From this assessment we can notice that FCM Designer, Mental Modeler and FCM Tool provide a suitable graphical support to the experts when analyzing scenarios and experimenting new situations, whereas JFCM is appropriate for developing FCM modules that could be reused in more complex solutions. However, these implementations still lack experimentation options and/or do not allow handling Machine Learning problems, which significantly hinder their usability when facing real-world situations. In contrast, FCM Expert and ISEMK stand as the most convenient software tools for developing FCM-based systems. The former is devoted to simulation and pattern classification scenarios whereas the latter is primarily focused on time series forecasting.

7 Concluding Remarks

In this review, we have surveyed relevant aspects of the FCM theory that include: (i) a walk through foundational concepts, with special emphasis on the dynamic properties of these systems as well as the different types of transfer functions and their bearing on convergence; (ii) the fundamental classes of FCM learning algorithms, ranging from Hebbian-based schemes to error-driven models and their amalgamations; (iii) the most recent FCM developments in the forecasting of univariate, multivariate and approximate time series as well as pattern classification scenarios and (iv) the list of available software tools and their practical considerations when designing FCM-based systems.

From the theoretical underpinnings of these *recurrent neural networks* we can state that selecting the proper transfer function and updating rule is a key aspect when designing the system. These factors will impact the network's capability of producing meaningful and accurate results.

One of the more prominent gaps in FCM theory lies in the learning algorithms to compute the weight set. As explained, Hebbian-based procedures have a poor generalization ability. On the other hand, the key drawback of the error-driven learning methods is that they provide solutions that are difficult to interpret. To the best of our knowledge, there is no error-driven procedure capable of producing authentic causal relations that match the modeled system. Discovering causal structures from historical records is quite challenging owing to the lack of statistical methods to measure causality. Other aspects such as the scalability or network convergence are also issues that remain open problems.

Regarding pattern classification, the prediction rates of low-level FCM-based classifiers are still often below those achieved by their black-box counterparts. We can conjecture about the reasons behind this behavior, for example, the absence of theoretically sound supervised learning algorithms with good generalization features. On the other hand, the prediction rates achieved by high-level FCM-based classifiers are indeed encouraging when compared to traditional black boxes yet a low-level analysis is no longer possible.

It should be highlighted that we can achieve different levels of interpretability depending on the abstraction degree used to model the FCM concepts. Entities with high abstraction level (i.e., information granules) lead to high-level interpretable models. If the level of abstraction is too high, then the network is difficult to analyze, but the reasoning process is still transparent. Conversely, defining attribute-level FCM concepts allows interpreting the system closer to its original physical/virtual representation. However, sometimes the experts are unable to define causal relations with such specificity level.

The results in time series forecasting using both attribute- and granular-level models are promising as well. Regrettably, low-level FCM-based forecasting models tend to converge to a fixed-point attractor when performing the inference process. While this feature is highly attractive in simulation and pattern classification situations, its effect on time series scenarios is less desirable. If the network converges to a fixed point, then the forecasting model will not fit the time series. This observation suggests that further investigation in this direction is required.

In this review study, we identified FCM Expert and ISEMK as the most complete tools for designing, learning and simulating FCM-based systems. FCM Expert includes algorithms for computing the weight matrix, reducing the network topology in dense FCM-based systems and improving the convergence without losing relevant information, while ISEMK comprises several learning methods for time series forecasting. Despite these facilities, both software products are still far from bridging the existing gap between theoretical advances in the FCM field and the development of sound practical applications.

Acknowledgment

The authors would like to thank Isel Grau (Vrije Universiteit Brussel, Belgium) and the anonymous reviewers for their valuable suggestions.

References

1. Abraham, A., Falcon, R., Bello, R.: Rough Set Theory: a True Landmark in Data Analysis. Springer Verlag, Berlin-Heidelberg, Germany (2009)
2. Aguilar, J., Contreras, J.: The FCM designer tool. In: Fuzzy Cognitive Maps, pp. 71–87. Springer (2010)
3. Ahmadi, S., Forouzideh, N., Alizadeh, S., Papageorgiou, E.: Learning fuzzy cognitive maps using imperialist competitive algorithm. Neural Computing and Applications **26**(6), 1333–1354 (2015)
4. Ahmadi, S., Forouzideh, N., Yeh, C.H., Martin, R., Papageorgiou, E.: A first study of fuzzy cognitive maps learning using cultural algorithm. In: Proceeding of the 2014 IEEE Conference on Industrial Electronics and Applications, pp. 2023–2028. IEEE (2014)
5. Alghzawi, A., Nápoles, G., Alghzawi, Vanhoof, K.: Forecasting social security revenues in jordan using fuzzy cognitive maps. In: Intelligent Decision Technologies, pp. 246–254. Springer (2018)
6. Alizadeh, S., Ghazanfari, M.: Learning FCM by chaotic simulated annealing. Chaos, Solitons & Fractals **41**(3), 1182–1190 (2009)
7. Alizadeh, S., Ghazanfari, M., Fathian, M.: Using data mining for learning and clustering FCM. International Journal of Computational Intelligence Systems **4**(2), 118–125 (2008)
8. Alizadeh, S., Ghazanfari, M., Jafari, M., Hooshm, S.: Learning FCM by tabu search. International Journal of Computer Science **2**(2), 142–149 (2007)
9. Amirkhani, A., Mosavi, M.R., Mohammadizadeh, F., Shokouhi, S.B.: Classification of intraductal breast lesions based on the fuzzy cognitive map. Arabian Journal for Science and Engineering **39**(5), 3723–3732 (2014)
10. Baran, R., Coughlin, J.: Convergence rates in symmetric neural networks with glauber dynamics. Mathematical and Computer Modelling **14**, 325–327 (1990)
11. Baran, R.H., Coughlin, J.P.: Simplified neuron model as a principal component analyzer. Journal of Mathematical Biology **15**, 267–273 (1982)
12. Baykasoglu, A., Durmusoglu, Z.D., Kaplanoglu, V.: Training fuzzy cognitive maps via extended great deluge algorithm with applications. Computers in Industry **62**(2), 187–195 (2011)
13. Bello, R., Falcon, R., Pedrycz, W., Kacprzyk, J.: Granular Computing: at the Junction of Rough Sets and Fuzzy Sets. Springer Verlag, Berlin-Heidelberg, Germany (2008)
14. Boutalis, Y., Kottas, T.L., Christodoulou, M.: Adaptive estimation of fuzzy cognitive maps with proven stability and parameter convergence. IEEE Transactions on Fuzzy Systems **17**(4), 874–889 (2009)
15. Bueno, S., Salmeron, J.L.: Benchmarking main activation functions in fuzzy cognitive maps. Expert Systems with Applications **36**(3), 5221–5229 (2009)
16. Buruzs, A., Hatwágner, M.F., Pozna, R.C., Kóczy, L.T.: Advanced learning of fuzzy cognitive maps of waste management by bacterial algorithm. In: 2013 Joint World Congress and NAFIPS Annual Meeting (IFSA/NAFIPS), pp. 890–895. IEEE (2013)
17. Carvalho, J.P., Tomé, J.A.: Qualitative optimization of fuzzy causal rule bases using fuzzy boolean nets. Fuzzy Sets and Systems **158**, 1931–1946 (2007)
18. Chen, Y., Mazlack, L., Lu, L.: Learning fuzzy cognitive maps from data by ant colony optimization. In: Proceedings of the 14th annual conference on Genetic and evolutionary computation, pp. 9–16. ACM (2012)
19. Chen, Y., Mazlack, L.J., Lu, L.J.: Inferring fuzzy cognitive map models for gene regulatory networks from gene expression data. In: Proceeding of the 2012 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), pp. 1–4. IEEE (2012)
20. Chen, Y., Mazlack, L.J., Minai, A.A., Lu, L.J.: Inferring causal networks using fuzzy cognitive maps and evolutionary algorithms with application to gene regulatory network reconstruction. Applied Soft Computing **37**, 667–679 (2015)
21. Chunmei, L., Yue, H.: Cellular automata learning of fuzzy cognitive map. In: Proceedings of the 2012 International Conference on System Science and Engineering (ICSSE), pp. 334–338. IEEE (2012)
22. De Franciscis, D.: JFCM: A Java library for fuzzy cognitive maps. In: Fuzzy Cognitive Maps for Applied Sciences and Engineering, pp. 199–220. Springer (2014)
23. Dickerson, J.A., Kosko, B.: Virtual worlds as fuzzy cognitive maps. Presence: Teleoperators & Virtual Environments **3**(2), 173–189 (1994)
24. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern classification, 2 edn. John Wiley & Sons (2012)

25. Froelich, W.: Towards improving the efficiency of the fuzzy cognitive map classifier. Neurocomputing **232**, 83–93 (2017)
26. Froelich, W., Juszczuk, P.: Predictive capabilities of adaptive and evolutionary fuzzy cognitive maps - a comparative study. In: Intelligent Systems for Knowledge Management, vol. 252, pp. 153–174. Springer (2009)
27. Froelich, W., Papageorgiou, E.I., Samarinas, M., Skriapas, K.: Application of evolutionary fuzzy cognitive maps to the long-term prediction of prostate cancer. Applied Soft Computing **12**(12), 3810–3817 (2012)
28. Froelich, W., Pedrycz, W.: Fuzzy cognitive maps in the modeling of granular time series. Knowledge-Based Systems **115**, 110–122 (2017)
29. Froelich, W., Salmeron, J.L.: Evolutionary learning of fuzzy grey cognitive maps for the forecasting of multivariate, interval-valued time series. International Journal of Approximate Reasoning **55**(6), 1319–1335 (2014)
30. Froelich, W., Salmeron, J.L.: Advances in fuzzy cognitive maps theory. Neurocomputing pp. 1–2 (2017)
31. Ghazanfari, M., Alizadeh, S., Fathian, M., Koulouriotis, D.E.: Comparing simulated annealing and genetic algorithm in learning FCM. Applied Mathematics and Computation **192**(1), 56–68 (2007)
32. Grau García, I., Nápoles, G.: Mutating HIV protease protein using ant colony optimization and fuzzy cognitive maps: drug susceptibility analysis. Computación y Sistemas **18**(1), 51–63 (2014)
33. Gray, S.A., Gray, S., Cox, L.J., Henly-Shepard, S.: Mental modeler: a fuzzy-logic cognitive mapping modeling tool for adaptive environmental management. In: Proceedings of the 46th Hawaii International Conference on System Sciences (HICSS), pp. 965–973. IEEE (2013)
34. Gregor, M., Groumpos, P.P.: Training fuzzy cognitive maps using gradient-based supervised learning. In: IFIP International Conference on Artificial Intelligence Applications and Innovations, pp. 547–556. Springer (2013)
35. Hagan, M.T., Menhaj, M.B.: Training feedforward networks with the marquardt algorithm. IEEE Transactions on Neural Networks **5**(6), 989–993 (1994)
36. Haykin, S.: Neural Networks: A Comprehensive Foundation, 2nd edn. Prentice Hall PTR, Upper Saddle River, NJ, USA (1998)
37. Hebb, D.O.: The organization of behavior: A neuropsychological theory. Psychology Press (1949)
38. Homenda, W., Jastrzebska, A., Pedrycz, W.: Joining concept's based fuzzy cognitive map model with moving window technique for time series modeling. In: Computer Information Systems and Industrial Management, pp. 397–408 (2014)
39. Homenda, W., Jastrzebska, A., Pedrycz, W.: Modeling time series with fuzzy cognitive maps. In: Proceedings of the 2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 2055–2062 (2014)
40. Homenda, W., Jastrzebska, A., Pedrycz, W.: Time Series Modeling with Fuzzy Cognitive Maps: Simplification Strategies, pp. 409–420. Springer (2014)
41. Homenda, W., Jastrzebska, A., Pedrycz, W.: Time series modeling with fuzzy cognitive maps: Simplification strategies - the case of a posteriori removal of nodes and weights. In: Computer Information Systems and Industrial Management, pp. 409–420 (2014)
42. Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. Proceedings of the National Academy of Sciences **79**, 2554–2558 (1982)
43. Huerga, A.V.: A balanced differential learning algorithm in fuzzy cognitive maps. In: Proceedings of the 16th International Workshop on Qualitative Reasoning, vol. 2002 (2002)
44. Kannappan, A., Papageorgiou, E.I.: A new classification scheme using artificial immune systems learning for fuzzy cognitive mapping. In: Proceedings of the 2013 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1–8. IEEE (2013)
45. Kannappan, A., Tamilarasi, A., Papageorgiou, E.I.: Analyzing the performance of fuzzy cognitive maps with non-linear Hebbian learning algorithm in predicting autistic disorder. Expert Systems with Applications **38**(3), 1282–1292 (2011)
46. Knight, C.J., Lloyd, D.J., Penn, A.S.: Linear and sigmoidal fuzzy cognitive maps: an analysis of fixed points. Applied Soft Computing **15**, 193–202 (2014)
47. Konar, A., Chakraborty, U.K.: Reasoning and unsupervised learning in a fuzzy cognitive map. Information Sciences **170**(2), 419–441 (2005)

48. Kosko, B.: Fuzzy cognitive maps. International Journal of man-machine studies **24**(1), 65–75 (1986)
49. Kosko, B.: Hidden patterns in combined and adaptive knowledge networks. International Journal of Approximate Reasoning **2**(4), 377–393 (1988)
50. Kosko, B.: Neural networks and fuzzy systems: a dynamical systems approach to machine intelligence. Prentice Hall, Upper Saddle River (1992)
51. Kottas, T., Boutalis, Y., Christodoulou, M.: Bi-linear adaptive estimation of fuzzy cognitive networks. Applied Soft Computing **12**(12), 3736–3756 (2012)
52. Kottas, T.L., Boutalis, Y.S., Christodoulou, M.A.: Fuzzy cognitive network: A general framework. Intelligent Decision Technologies **1**(4), 183–196 (2007)
53. Kottas, T.L., Boutalis, Y.S., Christodoulou, M.A.: Fuzzy cognitive networks: Adaptive network estimation and control paradigms. In: M. Glykas (ed.) Fuzzy Cognitive Maps: Advances in Theory, Methodologies, Tools and Applications, pp. 89–134. Springer (2010)
54. Koulouriotis, D., Diakoulakis, I., Emiris, D.: Learning fuzzy cognitive maps using evolution strategies: a novel schema for modeling and simulating high-level behavior. In: Proceedings of the 2001 Congress on Evolutionary Computation, vol. 1, pp. 364–371. IEEE (2001)
55. Kreinovich, V., Stylios, C.: Why fuzzy cognitive maps are efficient. Internationa Journal of Computer Communications & Control **10**(5), 825–833 (2015)
56. Kyriakarakos, G., Dounis, A.I., Arvanitis, K.G., Papadakis, G.: A fuzzy cognitive maps–petri nets energy management system for autonomous polygeneration microgrids. Applied Soft Computing **12**(12), 3785–3797 (2012)
57. León, M., Nápoles, G., Rodriguez, C., García, M.M., Bello, R., Vanhoof, K.: A fuzzy cognitive maps modeling, learning and simulation framework for studying complex system. In: New Challenges on Bioinspired Applications, pp. 243–256. Springer (2011)
58. Li, S.J., Shen, R.M.: Fuzzy cognitive map learning based on improved nonlinear Hebbian rule. In: Proceedings of the 2004 International Conference on Machine Learning and Cybernetics, vol. 4, pp. 2301–2306. IEEE (2004)
59. Lin, C., Chen, K., He, Y.: Learning fuzzy cognitive map based on immune algorithm. WSEAS Transactions on Systems **6**(3), 582–588 (2007)
60. Lu, W., Yang, J., Liu, X., Pedrycz, W.: The modeling and prediction of time series based on synergy of high-order fuzzy cognitive map and fuzzy c-means clustering. Knowledge-Based Systems **70**(70), 242–255 (2014)
61. Lu, W., Yang, J., Liui, X.: Numerical prediction of time series based on FCMs with information granules. International Journal of Computers Communications & Control **9**(3), 313–324 (2014)
62. Luo, X., Wei, X., Zhang, J.: Game-based learning model using fuzzy cognitive map. In: Proceedings of the first ACM international workshop on Multimedia technologies for distance learning, pp. 67–76. ACM (2009)
63. Madeiro, S.S., Von Zuben, F.J.: Gradient-based algorithms for the automatic construction of fuzzy cognitive maps. In: Proceedings of the 11th International Conference on Machine Learning and Applications (ICMLA), vol. 1, pp. 344–349. IEEE (2012)
64. Mateou, N.H., Moiseos, M., Andreou, A.S.: Multi-objective evolutionary fuzzy cognitive maps for decision support. In: Proceedings of the 2005 Congress on Evolutionary Computation, vol. 1, pp. 824–830. IEEE (2005)
65. McCulloch, W.S., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. In: J.A. Anderson, E. Rosenfeld (eds.) Neurocomputing: Foundations of Research, pp. 15–27. MIT Press (1988)
66. Miao, Y., Liu, Z.Q.: On causal inference in fuzzy cognitive maps. IEEE Transactions on Fuzzy Systems **8**(1), 107–119 (2000)
67. Miao, Y., Liu, Z.Q., Siew, C.K., Miao, C.Y.: Dynamical cognitive network - an extension of fuzzy cognitive map. IEEE Transactions on Fuzzy Systems **9**(5), 760–770 (2001)
68. Mohr, S.: Software design for a fuzzy cognitive map modeling tool. Tensselaer Polytechnic Institute (1997)
69. Nápoles, G., Bello, R., Vanhoof, K.: Learning Stability Features on Sigmoid Fuzzy Cognitive Maps through a Swarm Intelligence Approach, pp. 270–277. Springer (2013)
70. Nápoles, G., Bello, R., Vanhoof, K.: How to improve the convergence on sigmoid fuzzy cognitive maps? Intelligent Data Analysis **18**(6S), S77–S88 (2014)
71. Nápoles, G., Concepción, L., Falcon, R., Bello, R., Vanhoof, K.: On the accuracy-convergence trade-off in sigmoid fuzzy cognitive maps. IEEE Transactions on Fuzzy Systems (submitted) (2017)

72. Nápoles, G., Falcon, R., Papageorgiou, E., Bello, R., Vanhoof, K.: Rough cognitive ensembles. International Journal of Approximate Reasoning **85**, 79–96 (2017)
73. Nápoles, G., Falcon, R., Papageorgiou, E.I., Vanhoof, K.: Partitive granular cognitive maps to graded multilabel classification. In: Proceedings of the 2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1363–1370 (2016)
74. Nápoles, G., Grau, I., Bello, R., Grau, R.: Two-steps learning of fuzzy cognitive maps for prediction and knowledge discovery on the HIV-1 drug resistance. Expert Systems with Applications **41**(3), 821–830 (2014)
75. Nápoles, G., Grau, I., Falcon, R., Bello, R., Vanhoof, K.: A Granular Intrusion Detection System using Rough Cognitive Networks, pp. 169–191. Springer (2016)
76. Nápoles, G., Grau, I., Leon, M., Vanhoof, K.: A fuzzy cognitive maps tool for scenario analysis and pattern recognition. In: Proceedings of the 29th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2017) (2017)
77. Nápoles, G., Grau, I., Papageorgiou, E., Bello, R., Vanhoof, K.: Rough cognitive networks. Knowledge-Based Systems **91**, 46–61 (2016)
78. Nápoles, G., Grau, I., Vanhoof, K., Bello, R.: Hybrid model based on rough sets theory and fuzzy cognitive maps for decision-making. In: International Conference on Rough Sets and Intelligent Systems Paradigms, pp. 169–178. Springer (2014)
79. Nápoles, G., Mosquera, C., Falcon, R., Grau, I., Bello, R., Vanhoof, K.: Fuzzy-rough cognitive networks. Neural Networks (2017)
80. Nápoles, G., Papageorgiou, E., Bello, R., Vanhoof, K.: On the convergence of sigmoid fuzzy cognitive maps. Information Sciences **349–350**, 154–171 (2016)
81. Nápoles, G., Papageorgiou, E., Bello, R., Vanhoof, K.: Learning and convergence of fuzzy cognitive maps used in pattern recognition. Neural Processing Letters **45**, 431–444 (2017)
82. Oikonomou, P., Papageorgiou, E.I.: Particle swarm optimization approach for fuzzy cognitive maps applied to autism classification. In: IFIP International Conference on Artificial Intelligence Applications and Innovations, pp. 516–526. Springer (2013)
83. Papageorgiou, E., Aggelopoulou, K., Gemtos, T., Nanos, G.: Yield prediction in apples using fuzzy cognitive map learning approach. Computers and electronics in agriculture **91**, 19–29 (2013)
84. Papageorgiou, E., Spyridonos, P., Glotsos, D., Stylios, C.D., Ravazoula, P., Nikiforidis, G., Groumpos, P.P.: Brain tumor characterization using the soft computing technique of fuzzy cognitive maps. Applied Soft Computing **8**(1), 820–828 (2008)
85. Papageorgiou, E., Stylios, C.D., Groumpos, P.P.: Active Hebbian learning algorithm to train fuzzy cognitive maps. International journal of approximate reasoning **37**(3), 219–249 (2004)
86. Papageorgiou, E.I.: A new methodology for decisions in medical informatics using fuzzy cognitive maps based on fuzzy rule-extraction techniques. Applied Soft Computing **11**(1), 500–513 (2011)
87. Papageorgiou, E.I.: Learning algorithms for fuzzy cognitive maps - a review study. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) **42**(2), 150–163 (2012)
88. Papageorgiou, E.I., Froelich, W.: Multi-step prediction of pulmonary infection with the use of evolutionary fuzzy cognitive maps. Neurocomputing **92**, 28–35 (2012)
89. Papageorgiou, E.I., Groumpos, P.P.: Optimization of fuzzy cognitive map model in clinical radiotherapy through the differential evolution algorithm. Siomedical soft computing and human sciences **9**(2), 25–31 (2004)
90. Papageorgiou, E.I., Groumpos, P.P.: A new hybrid method using evolutionary algorithms to train fuzzy cognitive maps. Applied Soft Computing **5**(4), 409–431 (2005)
91. Papageorgiou, E.I., Groumpos, P.P.: A weight adaptation method for fuzzy cognitive map learning. Soft Computing **9**(11), 846–857 (2005)
92. Papageorgiou, E.I., Hatwágner, M.F., Buruzs, A., Kóczy, L.T.: A concept reduction approach for fuzzy cognitive map models in decision making and management. Neurocomputing **232**, 16–33 (2017)
93. Papageorgiou, E.I., Kannappan, A.: Fuzzy cognitive map ensemble learning paradigm to solve classification problems: application to autism identification. Applied Soft Computing **12**(12), 3798–3809 (2012)
94. Papageorgiou, E.I., Markinos, A.T., Gemtos, T.: Fuzzy cognitive map based approach for predicting yield in cotton crop production as a basis for decision support system in precision agriculture application. Applied Soft Computing **11**(4), 3643–3657 (2011)

95. Papageorgiou, E.I., Poczeta, K., Yastrebov, A., Laspidou, C.: Fuzzy Cognitive Maps and Multi-step Gradient Methods for Prediction: Applications to Electricity Consumption and Stock Exchange Returns, pp. 501–511. Springer (2015)
96. Papageorgiou, E.I., Poczta, K., Laspidou, C.: Hybrid model for water demand prediction based on fuzzy cognitive maps and artificial neural networks. In: Proceedings of the 2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1523–1530 (2016)
97. Papageorgiou, E.I., Salmeron, J.L.: A review of fuzzy cognitive maps research during the last decade. IEEE Transactions on Fuzzy Systems **21**(1), 66–79 (2013)
98. Papageorgiou, E.I., Salmeron, J.L.: Methods and algorithms for fuzzy cognitive map-based modeling. In: Fuzzy Cognitive Maps for Applied Sciences and Engineering, vol. 54, pp. 1–28. Springer (2014)
99. Papageorgiou, E.I., Stylios, C., Groumpos, P.P.: Unsupervised learning techniques for fine-tuning fuzzy cognitive map causal links. International Journal of Human-Computer Studies **64**(8), 727–743 (2006)
100. Papakostas, G., Koulouriotis, D.: Classifying patterns using fuzzy cognitive maps. In: Fuzzy Cognitive Maps, pp. 291–306. Springer (2010)
101. Papakostas, G.A., Boutalis, Y.S., E., K., Mertzios, B.G.: Fuzzy cognitive maps for pattern recognition applications. International Journal of Pattern Recognition and Artificial Intelligence **22**, 1461–1486 (2008)
102. Papakostas, G.A., Koulouriotis, D.E.: Classifying patterns using fuzzy cognitive maps. In: M. Glykas (ed.) Fuzzy Cognitive Maps: Advances in Theory, Methodologies, Tools and Applications, pp. 291–306. Springer Berlin Heidelberg (2010)
103. Papakostas, G.A., Koulouriotis, D.E., Polydoros, A.S., Tourassis, V.D.: Towards Hebbian learning of fuzzy cognitive maps in pattern classification problems. Expert Systems with Applications **39**(12), 10,620–10,629 (2012)
104. Papakostas, G.A., Koulouriotis, D.E., Polydoros, A.S., Tourassis, V.D.: Towards Hebbian learning of fuzzy cognitive maps in pattern classification problems. Expert Systems with Applications **39**(12), 10,620–10,629 (2012)
105. Parsopoulos, K.E., Papageorgiou, E.I., Groumpos, P., Vrahatis, M.N.: A first study of fuzzy cognitive maps learning using particle swarm optimization. In: Proceedings of the 2003 Congress on Evolutionary Computation, vol. 2, pp. 1440–1447. IEEE (2003)
106. Pawlak, Z.: Rough sets. International Journal of Computer & Information Sciences **11**(5), 341–356 (1982)
107. Pedrycz, W.: The design of cognitive maps: A study in synergy of granular computing and evolutionary optimization. Expert Systems with Applications **37**(10), 7288–7294 (2010)
108. Pedrycz, W., Homenda, W.: From fuzzy cognitive maps to granular cognitive maps. IEEE Transactions on Fuzzy Systems **22**(4), 859–869 (2014)
109. Penkova, T., Froelich, W.: Modeling and forecasting of well-being using fuzzy cognitive maps. In: Intelligent Decision Technologies 2016, pp. 241–250. Springer (2016)
110. Petalas, Y., Papageorgiou, E., Parsopoulos, K., Groumpos, P., Vrahatis, M.: Fuzzy cognitive maps learning using memetic algorithms. In: Proceedings of the international conference of Computational Methods in Sciences and Engineering(ICCMSE 2005), pp. 1420–1423 (2005)
111. Petalas, Y.G., Parsopoulos, K.E., Vrahatis, M.N.: Improving fuzzy cognitive maps learning through memetic particle swarm optimization. Soft Computing **13**(1), 77–94 (2009)
112. Poczketa, K., Yastrebov, A., Papageorgiou, E.I.: Learning fuzzy cognitive maps using structure optimization genetic algorithm. In: 2015 Federated Conference on Computer Science and Information Systems (FedCSIS), vol. 5, pp. 547–554. IEEE (2015)
113. Ren, Z.: Learning fuzzy cognitive maps by a hybrid method using nonlinear Hebbian learning and extended great deluge algorithm. In: Proceedings of the 23rd Midwest Artificial Intelligence and Cognitive Science Conference, pp. 159–163 (2012)
114. Salmeron, J.L.: Modelling grey uncertainty with fuzzy grey cognitive maps. Expert Systems with Applications **37**, 7581–7588 (2010)
115. Salmeron, J.L., Froelich, W.: Dynamic optimization of fuzzy cognitive maps for time series forecasting. Knowledge-Based Systems **105**, 2937 (2016)
116. Salmeron, J.L., Papageorgiou, E.I.: Fuzzy grey cognitive maps and nonlinear Hebbian learning in process control. Applied intelligence **41**(1), 223–234 (2014)
117. Senniappan, V., Subramanian, J., Papageorgiou, E.I., Mohan, S.: Application of fuzzy cognitive maps for crack categorization in columns of reinforced concrete structures. Neural Computing and Applications (2016)

118. Song, H., Miao, C., Roel, W., Shen, Z., Catthoor, F.: Implementation of fuzzy cognitive maps based on fuzzy neural network and application in prediction of time series. IEEE Transactions on Fuzzy Systems **18**(2), 233–250 (2010)
119. Song, H., Miao, C., Shen, Z., Roel, W., Maja, D., Francky, C.: Design of fuzzy cognitive maps using neural networks for predicting chaotic time series. Neural Networks **23**(10), 1264–1275 (2010)
120. Stach, W., Kurgan, L., Pedrycz, W.: A survey of fuzzy cognitive map learning methods. Issues in soft computing: theory and applications pp. 71–84 (2005)
121. Stach, W., Kurgan, L., Pedrycz, W.: Parallel learning of large fuzzy cognitive maps. In: International Joint Conference on Neural Networks, pp. 1584–1589. IEEE (2007)
122. Stach, W., Kurgan, L., Pedrycz, W.: Data-driven nonlinear Hebbian learning method for fuzzy cognitive maps. In: Proceedings of the 2008 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1975–1981. IEEE (2008)
123. Stach, W., Kurgan, L., Pedrycz, W.: A divide and conquer method for learning large fuzzy cognitive maps. Fuzzy Sets and Systems **161**(19), 2515–2532 (2010)
124. Stach, W., Kurgan, L., Pedrycz, W., Reformat, M.: Learning fuzzy cognitive maps with required precision using genetic algorithm approach. Electronics Letters **40**(24), 1519–1520 (2004)
125. Stach, W., Kurgan, L., Pedrycz, W., Reformat, M.: Genetic learning of fuzzy cognitive maps. Fuzzy sets and systems **153**(3), 371–401 (2005)
126. Stach, W., Kurgan, L.A., Pedrycz, W.: Numerical and linguistic prediction of time series with the use of fuzzy cognitive maps. IEEE Transactions on Fuzzy Systems **16**(1), 61–72 (2008)
127. Stylios, C.D., Groumpos, P.P.: Modeling complex systems using fuzzy cognitive maps. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans **34**(1), 155–162 (2004)
128. Tettamanzi, A.G., Tomassini, M.: Soft computing: integrating evolutionary, neural, and fuzzy systems. Springer (2013)
129. Tsadiras, A.K.: Comparing the inference capabilities of binary, trivalent and sigmoid fuzzy cognitive maps. Information Sciences **178**(20), 3880–3894 (2008)
130. Tsadiras, A.K., Margaritis, K.G.: An experimental study of the dynamics of the certainty neuron fuzzy cognitive maps. Neurocomputing **24**, 95–116 (1999)
131. Vanhoenshoven, F., Nápoles, G., Bielen, S., Vanhoof, K.: Fuzzy cognitive maps employing arima components for time series forecasting. In: Intelligent Decision Technologies, pp. 255–264. Springer (2018)
132. Wang, L., Pichler, E.E., Ross, J.: Oscillations and chaos in neural networks: an exactly solvable model. Proceedings of the National Academy of Sciences **87**(23), 9467–9471 (1990)
133. Yanchun, Z., Wei, Z.: An integrated framework for learning fuzzy cognitive map using RCGA and NHL algorithm. In: 4th International Conference on Wireless Communications, Networking and Mobile Computing (2008)
134. Yao, Y.: Three-way decisions with probabilistic rough sets. Information Sciences **180**(3), 341–353 (2010)
135. Yesil, E., Ozturk, C., Dodurka, M.F., Sakalli, A.: Fuzzy cognitive maps learning using artificial bee colony optimization. In: Proceedings of the 2013 IEEE international conference on fuzzy systems (FUZZ-IEEE), pp. 1–8. IEEE (2013)
136. Yesil, E., Urbas, L.: Big bang–big crunch learning method for fuzzy cognitive maps. World Academy of Science, Engineering and Technology **71**, 815–824 (2010)
137. Zhou, X., Zhang, H.: An algorithm of text categorization based on similar rough set and fuzzy cognitive map. In: Proceedings of the 5th International Conference on Fuzzy Systems and Knowledge Discovery, vol. 3, pp. 127–131. IEEE (2008)